

Introduction

This exercise consists of 2 sections:

1. How to Prepare Data for Tableau with FME (Processing Spreadsheets) (Current Article)

In Section 1, we'll process a single CSV file and perform data validation and cleanup before loading it into the Data Inspector

2. How to Prepare Data for Tableau with FME (Merging Multiple Spreadsheets)

In Section 2, we'll modify the FME workspace from Section 1 to handle multiple CSV files.

Step-by-Step Instructions

Part 1: Simple Translation

We will start by creating a simple translation, look at the results in Tableau, then come back to FME to do the data validation and cleanup.

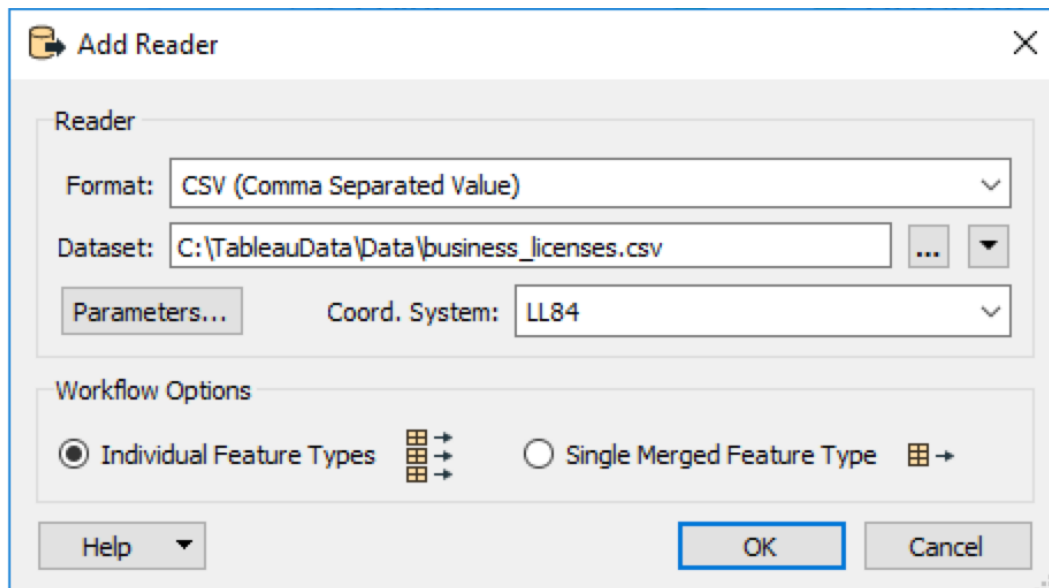
1. Create New Workspace

Start FME Workbench and select the New option under Create Workspace.

2. Set Up CSV Reader

The first step is to read the CSV file. Drag the business license CSV file from the file explorer onto the blank canvas. Notice, that FME has already filled in the reader format and dataset. Alternatively, click the Add Reader on the toolbar.

Reader Format:	CSV (Comma Separated Value)
Reader Dataset:	...\business_licenses.csv
Coord. System:	LL84



Add a CSV Reader, add the business_licenses.csv, set the coord system, then click on Parameters...

Click on the Parameters button. The Database Parameter allows us to choose different naming schemes for the layers or feature types that end up on the canvas. **Make sure it is set to "Feature Type from File Names".**

Confirm that the Attribute Definition has the Longitude and Latitude attributes set to x_coord and y_coord, respectively. By default, FME will map X/Y or Latitude/Longitude attributes to the x_coord and y_coord data types.

CSV (Comma Separated Value) Parameters

Dataset Parameters

Feature Type Name(s): From File Name(s)

Fields

Delimiter Character: auto

Field Names Line: 1

Data Start Line: 2

Advanced

Preview

	LicenceRSN	LicenceNumber	LicenceRevisionNumber	
2	2596895	16-138069	0	Korda Bros Decorat
3	2681995	16-113285	1	F & M Installations
4	2573412	16-114599	0	Battlefield Fight Le
5	2566683	16-107873	0	W Pile Construction
6	2657989	16-182604	0	Baron Woodwork Ir
7	2603391	16-144554	0	Lifelike Stylz Rental

Attributes

Attribute Definition

☒ Automatic ☐ Manual

Read	Name	Type	Width	Precision
☒	Latitude	y_coordinate		
☒	Longitude	x_coordinate		
☒	FeePaid	uint32		

☐ Select all Filter ☐ Select All

Schema Attributes

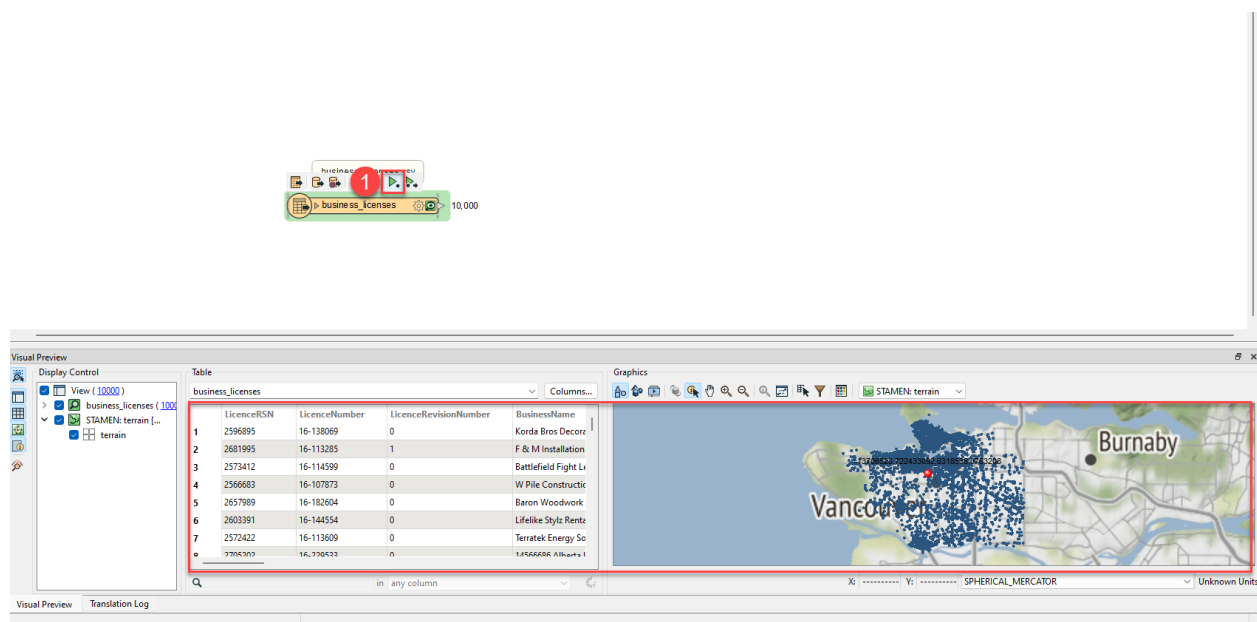
Help Presets OK Cancel

If necessary, change the Attribute Definition to Manual and update the Latitude and Longitude

Click Ok, and then click Ok again to add the Reader to the canvas. FME is able to create point geometry for each record with a latitude and longitude value.

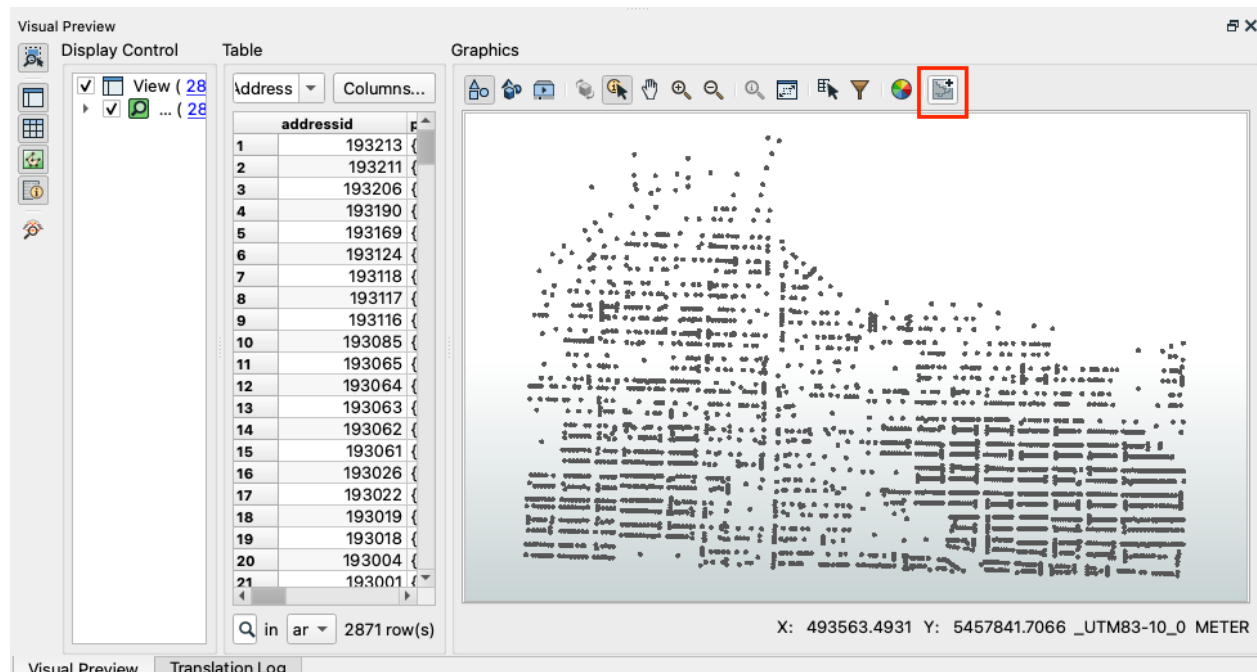
3. Confirm Points are Created as CSV File is Read

Once the feature type has been added to the canvas, select the business_licenses feature type and click the run to this button. This will run the translation to the selected object and generate a **feature cache** which can be inspected in the visual preview window or the Data Inspector by clicking on the green feature cache icon. We can see that we do, in fact, have points.

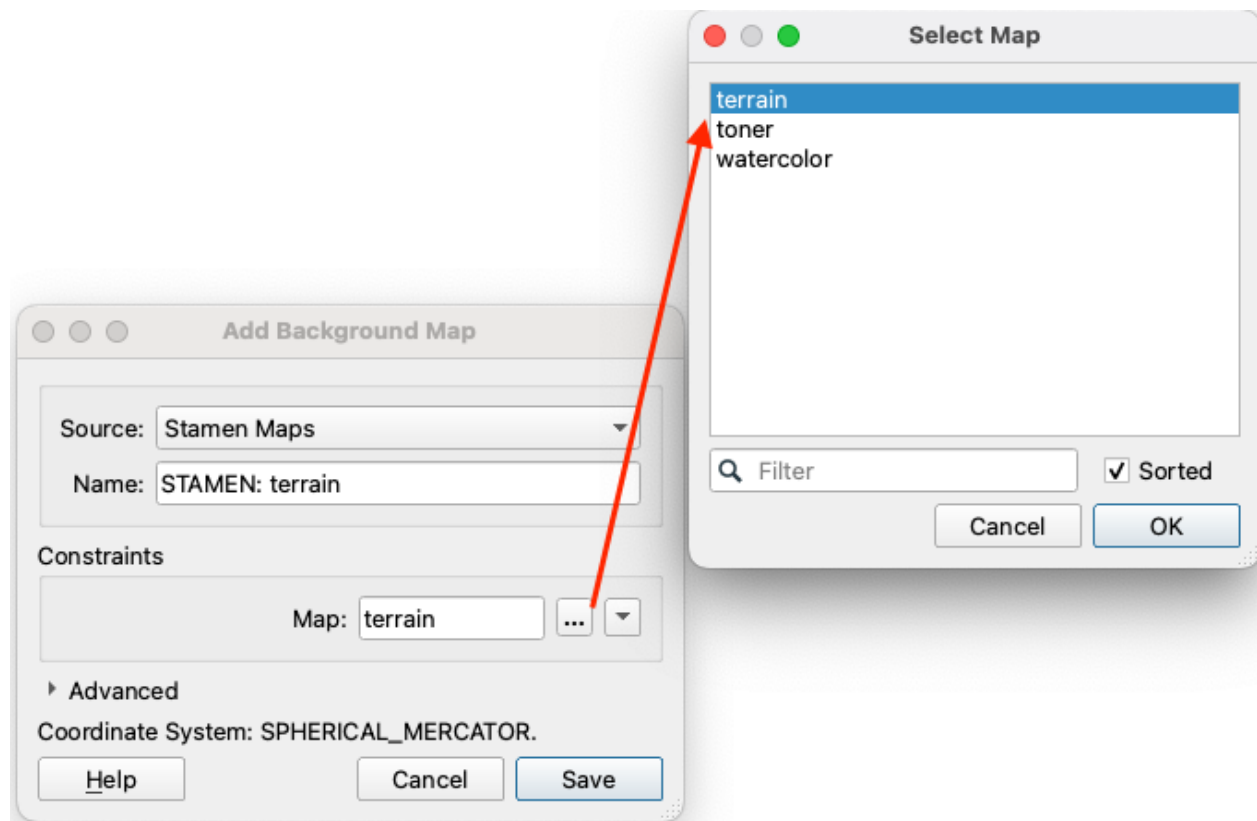


Business Licenses with point data in the Visual Preview Window

If you are missing the background map, you can easily add it using the add a background map button on the Visual Preview toolbar.



Here you can choose from a variety of background map services. Select the Stamen Maps: Terrain background map and then click Save to apply it.



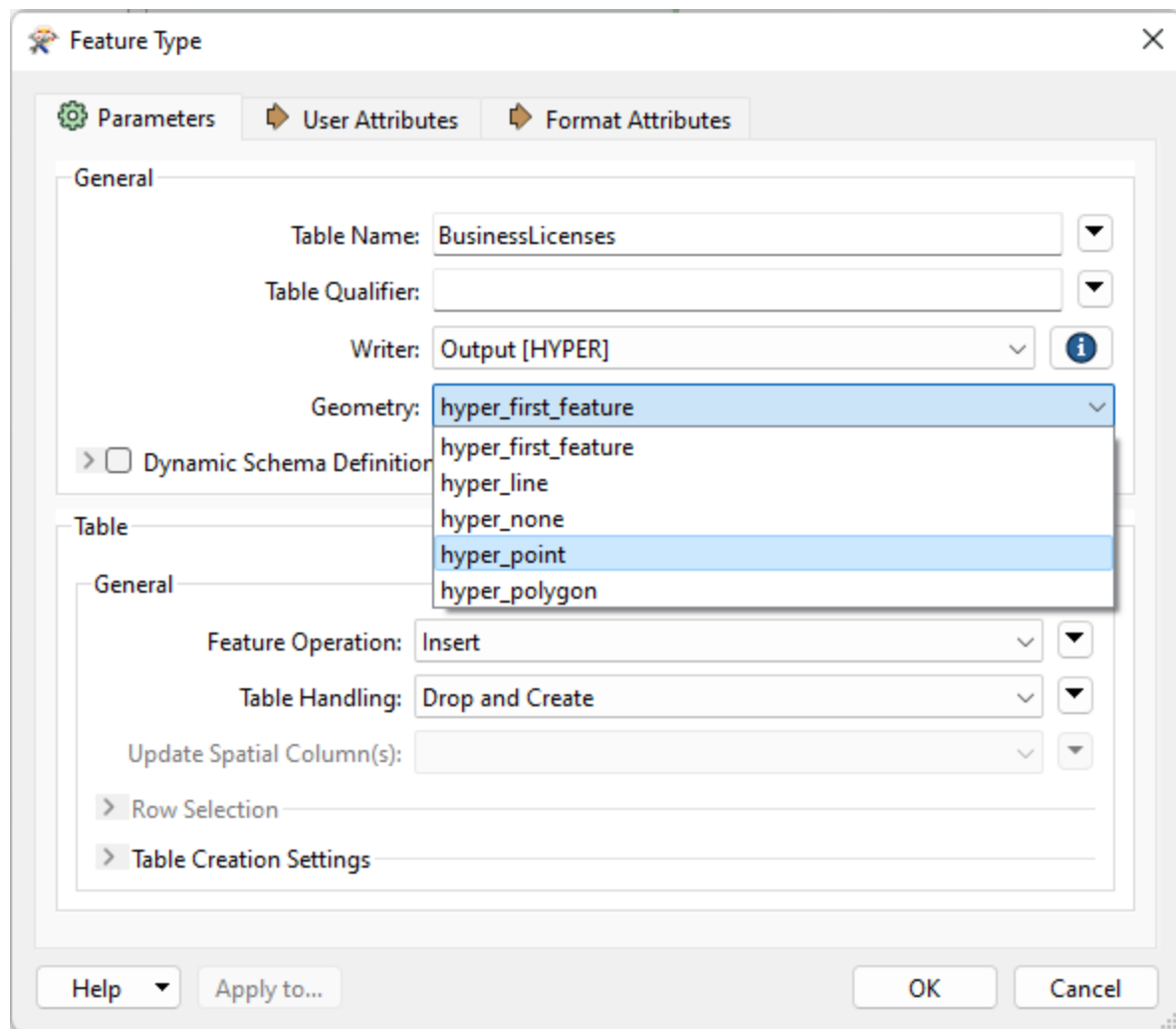
4. Add Tableau Writer

From the Writers menu, select Add Writer or using the Quick Add Menu, start typing Tableau, and select the Tableau Hyper format. For Dataset, specify a directory in which to write the .hyper file. Lastly, set the Table Definition to Automatic then click OK to add the Tableau writer to the workspace. Then connect the Reader to the Writer.



5. Modify Writer Feature Type Properties

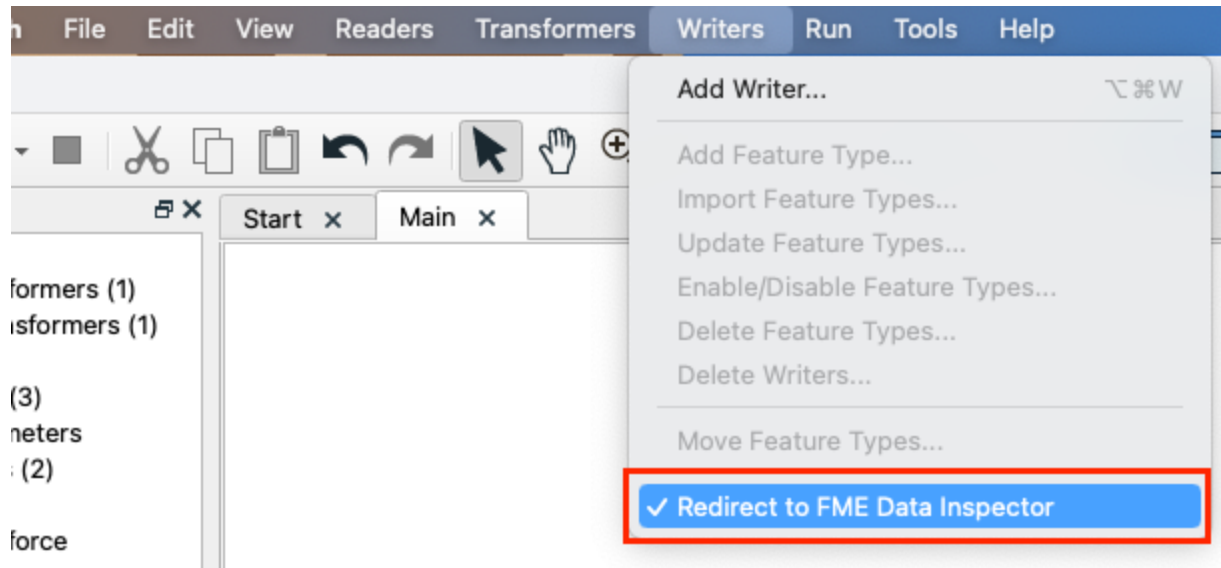
Open the properties of the Writer. We can now specify the name of the table we would like to write to. Call it BusinessLicenses.



Change the Writer Feature Type Properties by changing the Table Name

6. Run Workspace

Before running the workspace, enable the Redirect to FME Data Inspector option in the Menu > Writers > Redirect to FME Data Inspector (should have a check mark). *We are enabling this so we can preview the output directly in the Data Inspector as we do not have Tableau licenses on the training machines and because not all attendees are Tableau users.*



With Redirect to FME Data Inspector enabled, click the Run button to run the workspace.

You may have noticed a number of blue warnings go by in the log file. These are related to problems with the CSV data, which we will soon fix with FME.

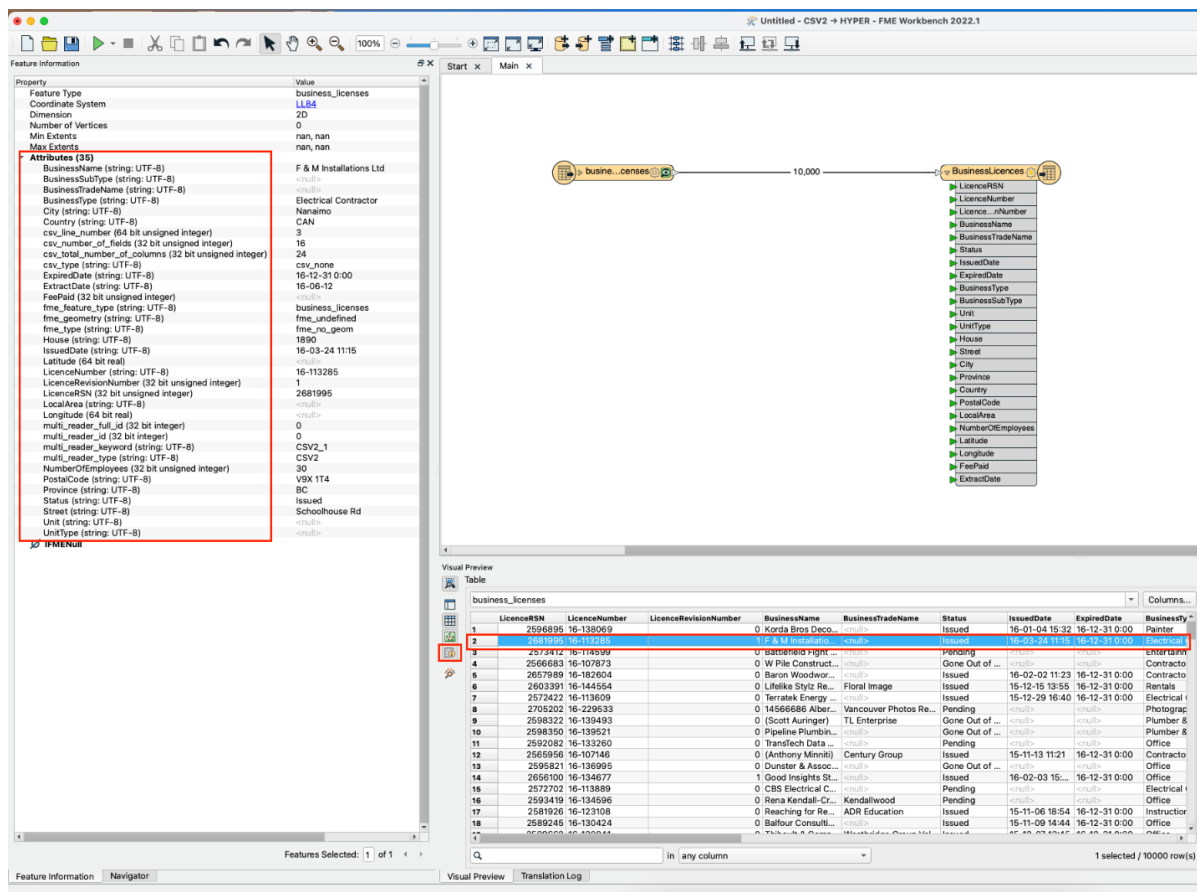
7. Examine Output in the Visual Preview Window

In the Visual Preview window, we can see that all of the columns correspond to the columns we read in from the CSV file with FME. Notice that the data types of the columns have all been set automatically by FME based on the best guess at what kind of data is inside each one. If desired, you can manually set the data types, width and precision in the writer feature type on the User Attributes tab by changing the Attribute Definition to Manual.

If we were to open the output in Tableau, the result would look like the image below where we can see that the LicenseRSN has correctly been set to a Number type and BusinessName is a String.

#	Abc	Abc
Extract	Extract	Extract
LicenceRSN	LicenceNumber	BusinessName
2680780	16-205238	Vivo Media Arts Centre
2588291	16-129470	North Arm Transport...
2588257	16-129436	J Mcleod
2588030	16-129209	F D G Management Ltd

View the data in Visual Preview to ensure it has set the correct data types. You can view data types by enabling the Feature Information Window button  on the Visual Preview toolbar.



The screenshot displays the FME Workbench 2022.1 interface. On the left, the 'Feature Information' window is open, showing a list of attributes for the 'business_licenses' feature type. The attributes are listed with their data types, such as 'BusinessName (string: UTF-8)', 'LicenceNumber (32 bit unsigned integer)', and 'Status (string: UTF-8)'. The 'Visual Preview' window at the bottom shows a table of data for the 'business_licenses' feature type. The table has columns for 'LicenceRSN', 'LicenceNumber', 'LicenceRevisionNumber', 'BusinessName', 'BusinessTradeName', 'Status', 'IssuedDate', 'ExpiresDate', and 'BusinessType'. The data is filtered to show 1 selected row out of 10000 rows.

LicenceRSN	LicenceNumber	LicenceRevisionNumber	BusinessName	BusinessTradeName	Status	IssuedDate	ExpiresDate	BusinessType
2680780	16-205238	1	Vivo Media Arts Centre		Issued	16-01-04 15:32	16-12-31 0:00	Painter
2588291	16-129470	1	North Arm Transport...		Issued	16-03-24 11:15	16-12-31 0:00	Electrical
2588257	16-129436	1	J Mcleod		Issued	16-03-24 11:15	16-12-31 0:00	Electrical
2588030	16-129209	1	F D G Management Ltd		Issued	16-03-24 11:15	16-12-31 0:00	Electrical

Part 2: Data Validation and Cleanup

Let's take a closer look at the business license data.

Here is a list of tasks we will accomplish with FME. We have already completed the first one in our current workspace. We will modify the workspace to perform the rest of the tasks.

- Read Excel file and create points
- Filter out records that don't have latitude/longitude values
- Ensure PostalCode has a value; extract first 3 characters
- Set up conditional value to handle empty FeePaid values
- Create new BusinessDisplayName field; populate from existing fields

1. Filter out records that don't have latitude/longitude values

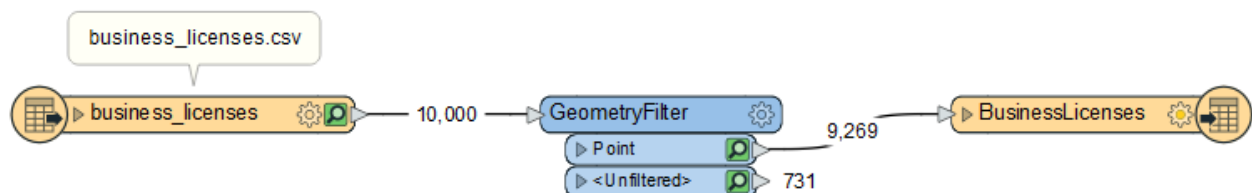
Some of the records in the CSV file did not have latitude and longitude values. Since FME would not have been able to create points for them, we want to filter them out; a [GeometryFilter](#) will help us accomplish this.

Click on the connection between the CSV Reader Feature Type and the Writer Feature Type and then start typing "GeometryFilter" click enter. Click enter again to enter the Transformer Parameters. Leave the Mode as Simple and select "Point" for the "Output Types". Then click Ok to accept the parameters.

2. Run workspace with feature caching enabled

Run the workspace to generate feature caches on the [GeometryFilter](#). If it's not already enabled, you can turn feature caching on in the Run submenu > Enable Feature Caching

We can confirm that 731, out of our 10,000 records did not have latitude or longitude values because no point geometry was created for these features.

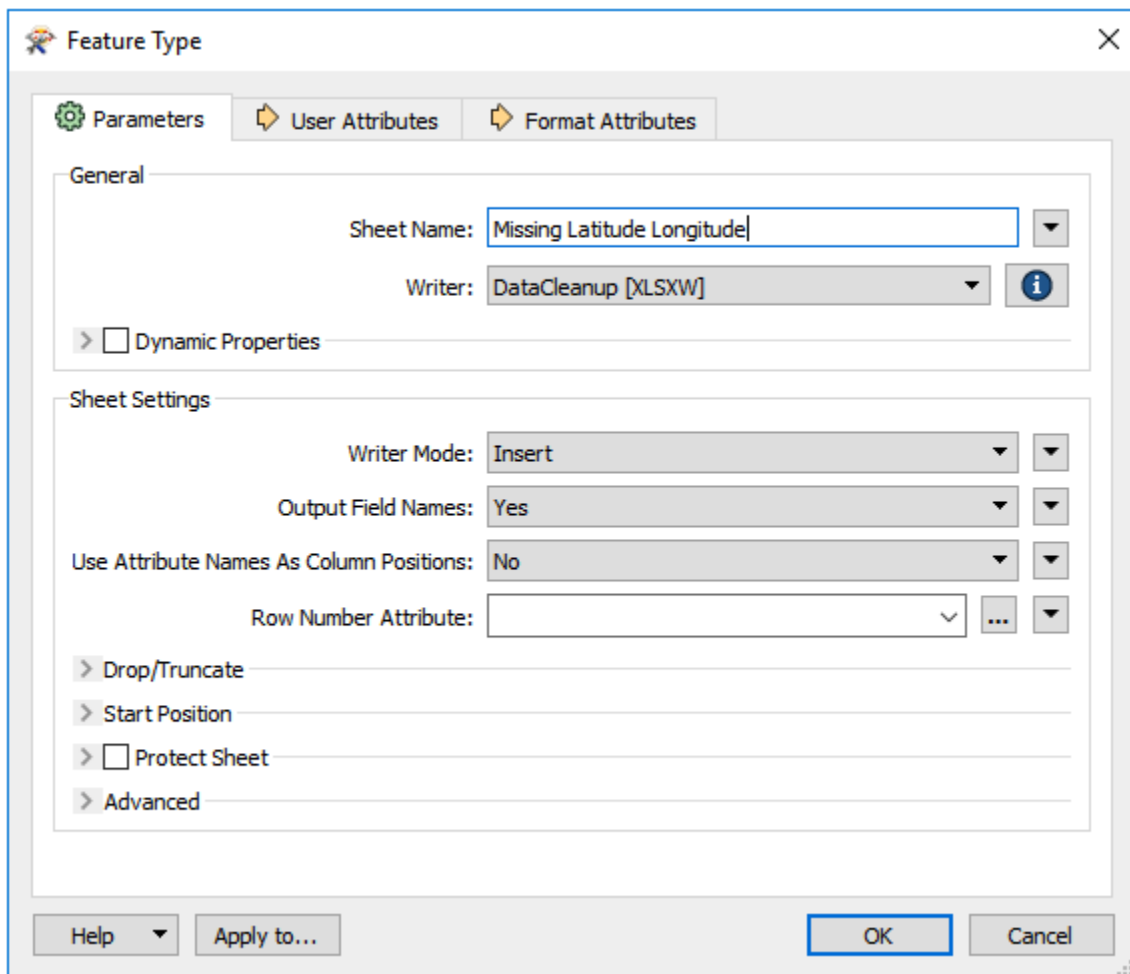


Workspace with feature caching turned on to view <Unfiltered> count

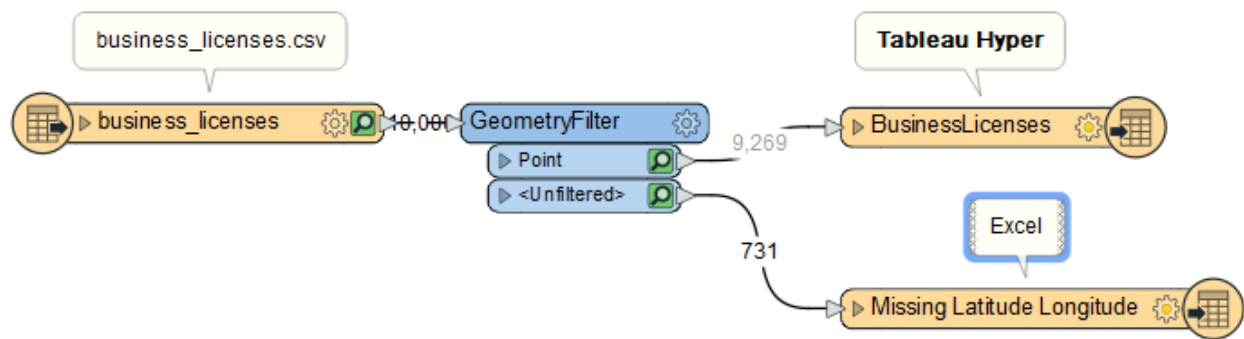
3. Write records with no geometry to "Data Cleanup" Excel file

Before we continue processing the points, let's write these records out to a "Data Cleanup" file, so that they may be fixed. We will write them out to Excel.

From the Writers menu, add an Excel writer. Write it to the output folder and **call the file DataCleanup.xlsx**, and **set the Sheet Definition to Automatic** then click Ok to add the writer to the canvas. Connect the new writer feature type up to the unfiltered port of the GeometryFilter, then open up the Writer's properties and change the sheet name to "Missing Latitude Longitude".



Change the name of the sheet to identify what we are recording



We will return to this stream of features later on in the exercise, but for now, let's continue processing the points.

4. Ensure PostalCode has values: AttributeValidator

Place an **AttributeValidator** on the canvas and connect it to GeometryFilter's Point output port. Open its properties and select PostalCode for the Attribute to Validate. The validation rule is that PostalCode "Has a Value". Also, validate that some of the string fields have string values and that the numeric fields have numeric values. For example, Country & City should be String data types, and NumberOfEmployees, FeePaid should be Numeric.

AttributeValidator Parameters

Transformer

Transformer Name:

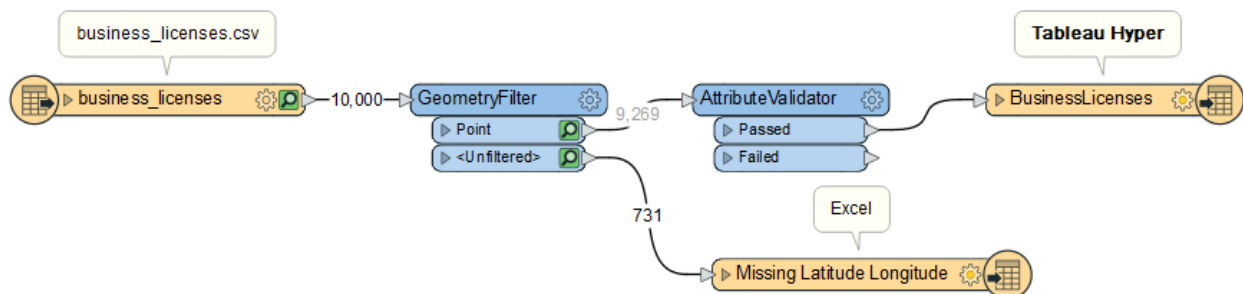
Validation Rules

Attributes to Validate	Validation Rule	Rule Configuration
PostalCode	Has a Value	<Unused>
Country,City	Type	String
NumberOfEmployees,Fe...	Type	Numeric

+ - ▲ ▼ ≡

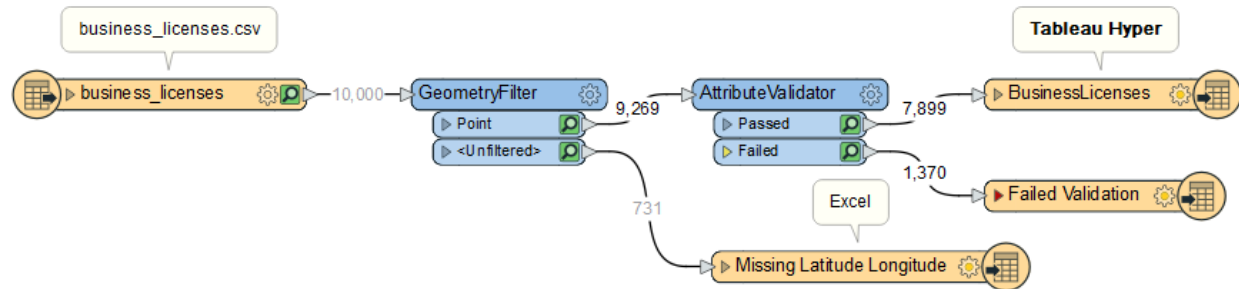
Help Defaults OK Cancel

Set the AttributeValidator to validate if PostalCode has a value and other attributes are the proper type



5. Write records that fail validation to new sheet in "Data Cleanup" Excel file

Again, before continuing to process the valid data, write the data that fails validation out to a different sheet in the "Data Cleanup" excel file already set up. Right-click on the canvas and select "Insert Writer Feature Type". Call the new sheet "Failed Validation". Connect it to the Failed port of the AttributeValidator.



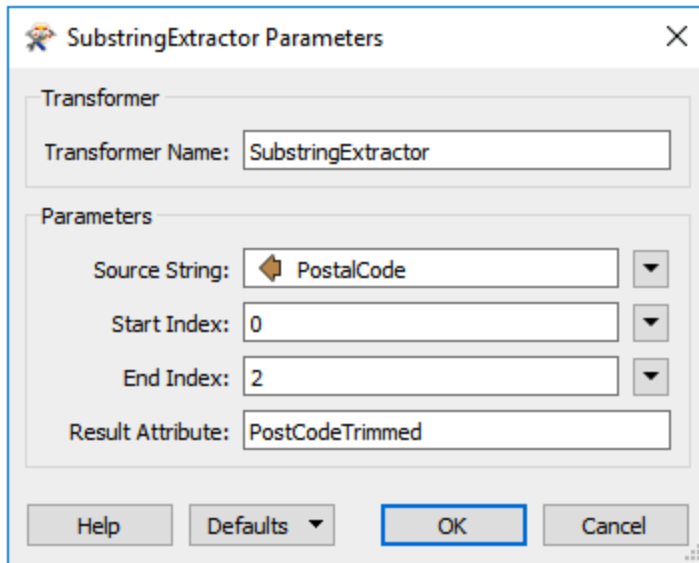
6. Extract first 3 characters of PostalCode: SubStringExtractor

Now that we know that the "records output" from the AttributeValidator all have a value for Postal Code, let's extract the first 3 characters. We do this because Tableau uses the first 3 characters of the postal code to automatically map the areas.

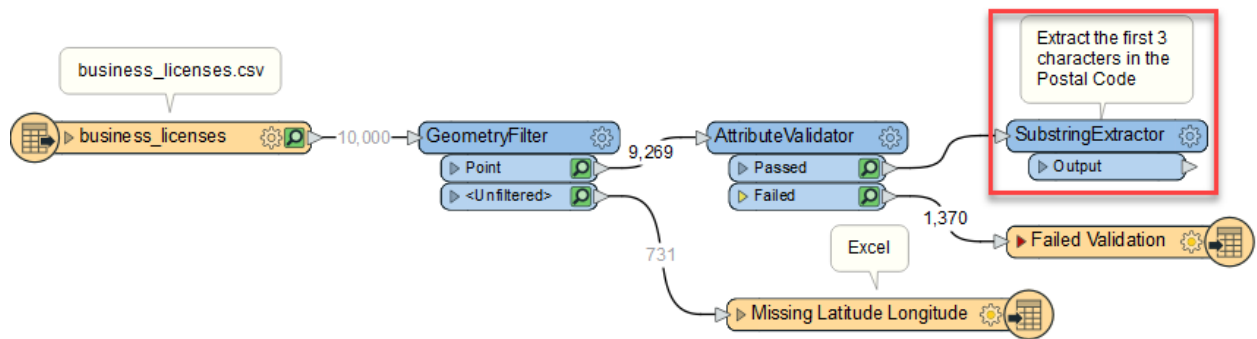
Place a **SubStringExtractor** on the canvas, connect it to the Passed port of the AttributeValidator, and configure it to extract the first 3 characters from the PostalCode Attribute. We will call the resulting attribute PostCodeTrimmed.

FME indexes the first value as 0 then counts from there. As you can see in the table below, if we only want the first 3 characters in the postal code, we would go from 0 to 2. So in the SubstringExtractor our Start Index = 0 and our End Index = 2.

V	3	W	<Space>	1	J	8
0	1	2	3	4	5	6

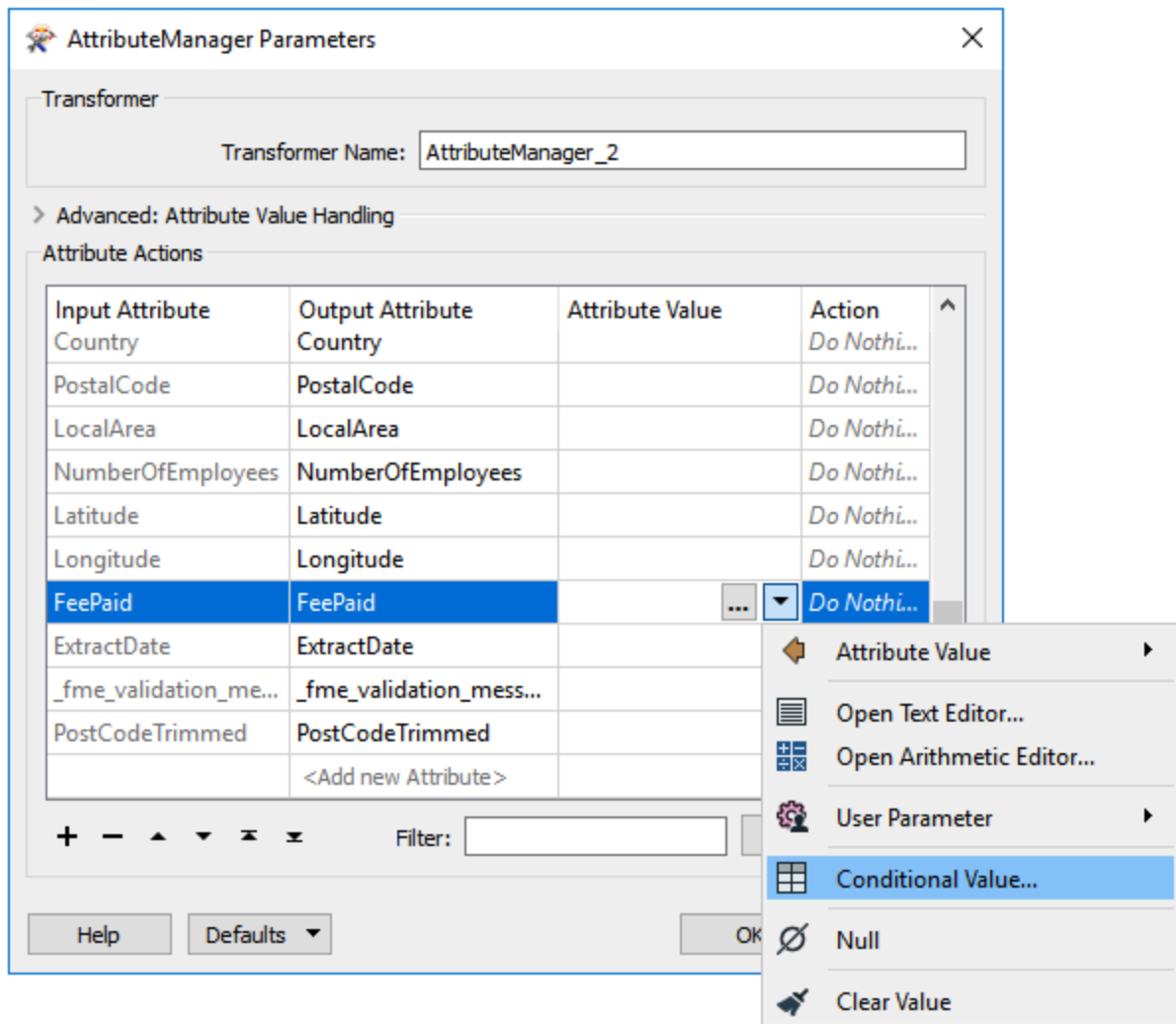


Setup the SubstringExtractor to extract the first 3 values from PostalCode



7. Set up conditional attribute values for FeePaid: AttributeManager

The **AttributeManager** is a transformer that allows us to do many attribute manipulations, including setting up **conditional values**. Add the AttributeManager after the SubstringExtractor



Create a Conditional Value in the AttributeManager for FeePaid

If the FeePaid attribute is empty, set the value to 0, otherwise, leave it as is.

Parameter Condition Definition

Condition Statement

	Test Condition	Attribute Value
If	FeePaid ATTRIBUTE_IS_EMPTY	☐ 0.0
Else If		
Else	<All Other Conditions>	☒ <No Action>

+ - ▲ ▼ ✂ 📄 📋 Edit...

Help OK Cancel

Create a new attribute called BusinessDisplayName. Set its value to BusinessTradeName, but only if BusinessTradeName has a value, otherwise set it to BusinessName.

Parameter Condition Definition

Condition Statement

	Test Condition	Output Value
If	BusinessTradeName ATTRIBUTE_HAS_VALUE	📌 BusinessTradeName
Else If		
Else	<All Other Conditions>	📌 BusinessName

+ - ▲ ▼ Edit... Duplicate

Help OK Cancel

If BusinessTradeName has an attribute Value, keep it BusinessTradeName otherwise name it BusinessName

AttributeManager Parameters

Transformer

Transformer Name: AttributeManager

> Advanced: Attribute Value Handling

Attribute Actions

Input Attribute	Output Attribute	Attribute Value	Action
NumberOfEmployees	NumberOfEmployees		Do Nothing
Latitude	Latitude		Do Nothing
Longitude	Longitude		Do Nothing
FeePaid	FeePaid	2 Possible Values	Set Value
ExtractDate	ExtractDate		Do Nothing
_fme_validation_messa...			Remove
PostCodeTrimmed	PostCodeTrimmed		Do Nothing
	BusinessDisplayName	2 Possible Values	Set Value
	<Add new Attribute>		

+ - ▲ ▼ ↕

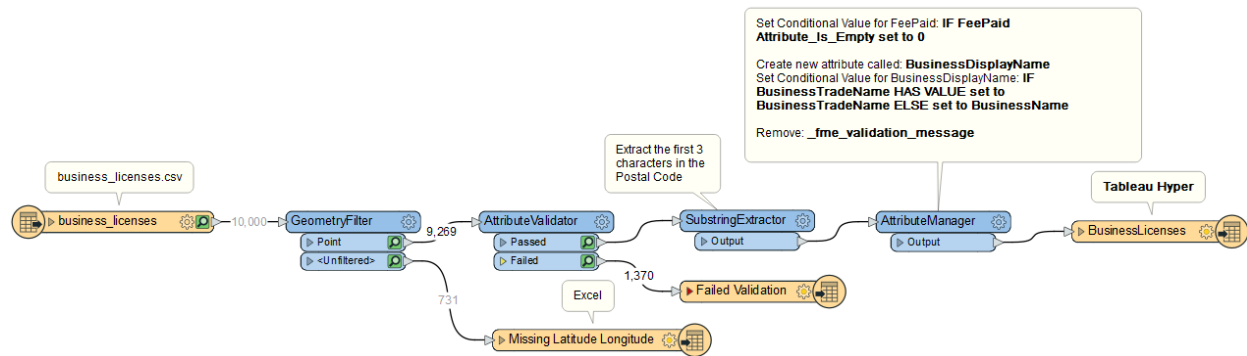
Filter:

Import ...

Help Defaults OK Cancel

In the AttributeManager, create 2 conditional values for FeePaid and BusinessDisplayName. This is also where you can re-order your attributes if you want your columns ordered in a particular way. Note: if you are changing the order of attributes and you are working with a schema that already exists, you may need to drop and create the table to invoke the changes.

Connect the AttributeManager output to the Writer feature type.



Your workspace should now look like the image above. Next we will work on the <Unfiltered> output port on the GeometryFilter.

8. Filter for features to Geocode

Since the GeometryFilter is filtering for features with point geometry, we know our unfiltered points are missing X (longitude) and Y (latitude) values. However, we also know that this dataset contains address information which we can use to geocode in order to obtain Latitude and Longitude values. Rather than simply connecting the <Unfiltered> features to a [Geocoder](#), we'll first filter out any features that do not contain address information.

Add the [Tester](#) after the GeometryFilter and connect it to the Unfiltered output port. Once connected, set the following Test Clauses to ensure attributes containing address information have a value. For example, City, Country, PostalCode, Street, AND House should have a value. *Note: the Logic parameter is set to AND to ensure each feature has a value for each of the attributes we have listed in the Test Clauses.*

Tester Parameters

Transformer Name:

Test Clauses

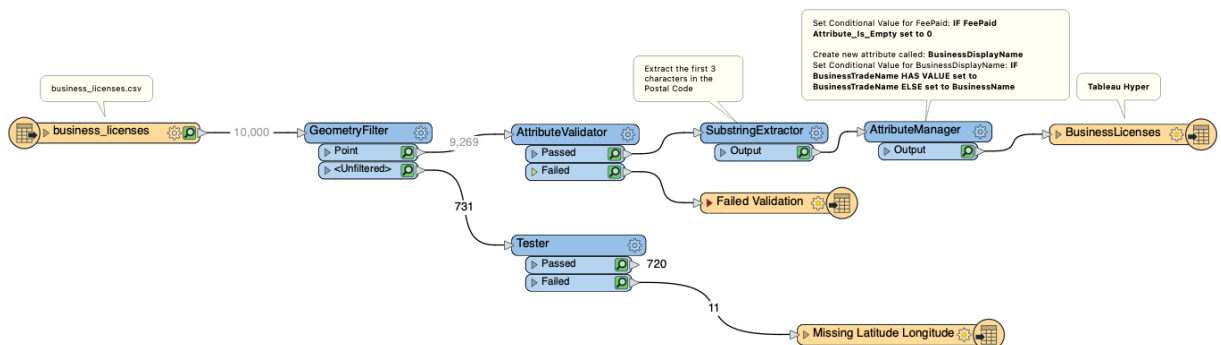
Logic	Left Value	Operator	Right Value
	City	Attribute Has a Value	<Unused>
AND	Country	Attribute Has a Value	<Unused>
AND	PostalCode	Attribute Has a Value	<Unused>
AND	Street	Attribute Has a Value	<Unused>
AND	House	Attribute Has a Value	<Unused>

Comparison Mode:

Composite Expression

Advanced

Since we want to geocode features that meet this criteria, the failed port can be connected to the Missing Latitude Longitude writer feature type (shown below) as these will need to be manually reviewed.



Since we are only geocoding a handful of features, we do not have to specify an API key.

safe.geocoder.Geocoder Parameters

Transformer Name:

Service

Geocoding Service:

Usage Information

No API key? Sign up for [Geocoder.ca](#).

Use of Geocoder.ca is subject to the [Geocoder.ca Terms of Service](#).

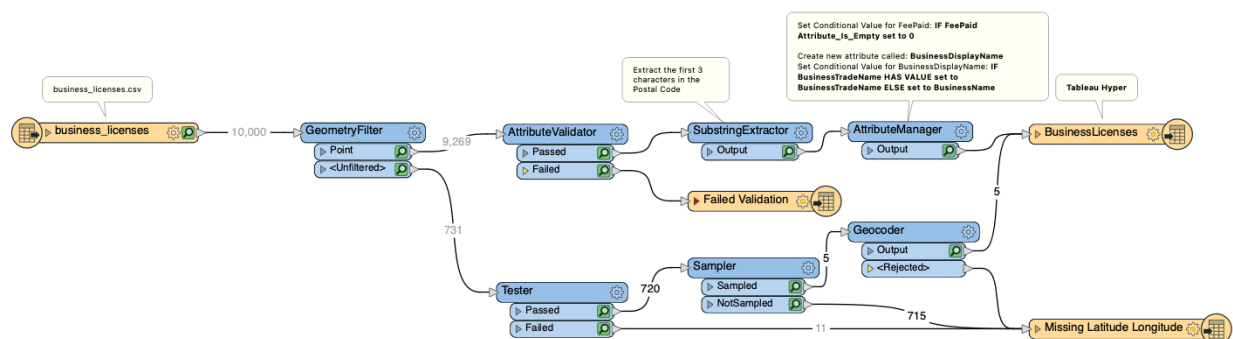
Connection

API Key:

Geocoding Parameters

Street Address:

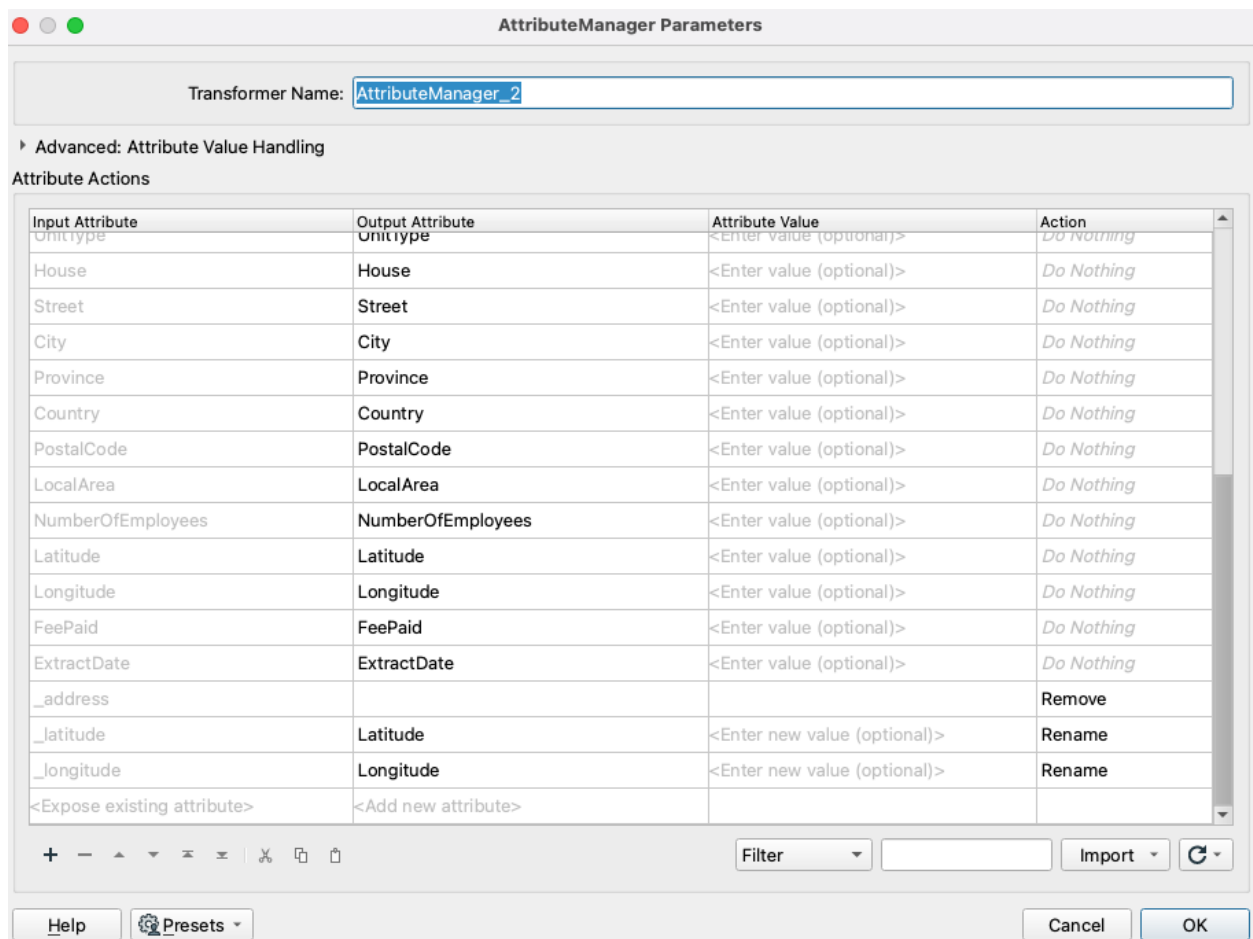
Advanced



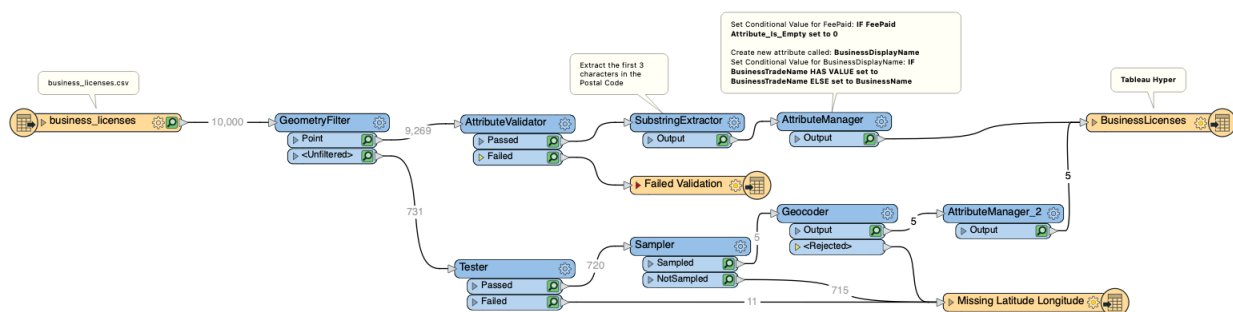
11. Remove unwanted attributes from the Geocoder

In many cases, the features that are successfully returned by the Geocoder will contain additional attributes. In this case, 3 new attributes have been added (`_address`, `_latitude`, and `_longitude`).

Add an AttributeManager after the Geocoder to **remove the _address** attribute and rename the **_latitude to Latitude** AND **_longitude to Longitude** to match the desired output schema. Lastly, connect the AttributeManager to the BusinessLicenses Writer FeatureType to complete the workspace.



Your workspace should look like similar to the image below:



12. Modify writer feature type properties

Since we defined an Automatic attribute definition, the schema on our Writer Feature Type has been automatically set based on the incoming connections to the Feature Type. If you want to update data types, re-order, or set any indexing rules you can do so by changing the attribute definition to Manual in the Writer Feature Type Parameters on the User Attributes tab.

Since we are creating a new Tableau Hyper file, we can leave the Table Handling as Create if needed.

13. Run workspace and confirm .hyper file exists

Let's run our final workspace! Click on the Run button and the output will automatically open in the Visual Preview window.

If you wish to create the actual files, you can disable the Redirect to FME Data Inspector setting in the Writers menu. If you have disabled it, run the workspace and confirm the .hyper file and DataCleanup.xlsx files were created using Windows Explorer.

In this exercise, you learned how to 1) filter by geometry type, data type, and attribute values, 2) extract portions of text using string indices (though you can also extract text with REGEX and other techniques as well), 3) geocode addresses, and 4) perform schema changes like adding, renaming, re-ordering and how to update data types.

In the next exercise, we will look at creating a few ways you can build density maps with FME for Tableau.